UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/621,446	07/18/2003	Un-gyo Jung	Q75479	1015

23373 7590 09/21/2007

SUGHRUE MION, PLLC
2100 PENNSYLVANIA AVENUE, N.W.
SUITE 800
WASHINGTON, DC 20037

EXAMINER	
LOVEL, KIMBERLY M	

ART UNIT	PAPER NUMBER
2167	

MAIL DATE	DELIVERY MODE
09/21/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/621,446	JUNG, UN-GYO	
	Examiner	Art Unit	
	Kimberly Lovel	2167	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *04 June 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-17* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-17 are pending.

Continued Examination Under 37 CFR 1.114

2. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 4 June 2007 has been entered.

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. Claims 7 is rejected under 35 U.S.C. 102(b) as being anticipated by US Patent No 6,289,506 to Kwong et al (hereafter Kwong et al).

Referring to claim 7, Kwong et al disclose a method of improving the performance of a Java platform by executing a Java application on the Java platform in a device, the Java application causing the device to perform a desired function, (see abstract), the method comprises:

(a) precompiling a class file included in a standard class library into an extended class library file including a machine instruction (see column 5, lines 8-67);

(b) the extended class library file executing the machine instruction (see column 5, lines 38-67 and column 3, lines 44-47); and

(c) executing a Java application file by using at least one of a Just-In-Time (JIT) compiling method and an interpreting method (see column 3, lines 37-64).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-3, 14, 15 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over US PGPub 2003/0005425 to Zee (hereafter Zee) in view of US PGPub 2002/0059475 to Baentsch et al.

Referring to claim 1, Zee discloses a Java execution device which is executed on a Java platform (see abstract) comprising:

an extended class library which includes a class file of a machine code obtained by precompiling a class file [Java class file] included in a standard class library (see [0060], line 6 – [0064] – an AOT compiler is utilized to compile the Java class file, which is then stored in a database until the compiled file is requested by processing system 30 or 32); and

a Java Virtual Machine (JVM) [data processing systems 30 and 32] which executes the class file of the machine code class file or an application file included in the extended class library (see [0037] and [0041]).

However, Zee fails to explicitly disclose wherein data processing systems 30 and 32 contain a Java Virtual Machine. Baentsch et al disclose a Java run-time system with modified linking identifiers (see abstract), including the use of a Java Virtual Machine foe executing class files (see [0005]).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the Java Virtual Machine disclosed by Baentsch et al as a software-only platform running on top of the hardware-based platform of processing systems 30 and 32 disclosed by Zee. One would have been motivated to do so since the processing systems of Zee have the ability to execute a Java class file made of a Java virtual machine language of bytecodes that has been compiled (Zee: see [0037]).

Referring to claim 2, the combination of Zee and Baentsch et al (hereafter Zee/Baentsch) discloses the Java execution device of claim 1, wherein a machine

instruction of the machine code includes an operand in which symbolic reference information [references which name target items to which they refer] is inserted (Baentsch et al: see [0008]-[0009]).

Referring to claim 3, Zee/Baentsch discloses the Java execution device of claim 2, wherein the Java Virtual Machine (JVM) includes a class linker which converts the symbolic reference information inserted in the operand of the machine instruction into an address (Baentsch et al: see [0008], lines 11-15).

Referring to claim 14, Zee/Baentsch discloses the Java execution device of claim 1, wherein an Ahead-of-Time (AOT) compilation is performed [performed by AOT compiler] on the class file prior to execution of the class file by the Java Virtual Machine (JVM) [processing systems 30 and 32] (Zee: see [0062]-[0064] – the AOT compiler compiles the Java class file and stores the file in the database until the file is requested by processing systems 30 and 32).

Referring to claim 15, Zee/Baentsch discloses the Java execution device of claim 1, wherein a machine instruction of the machine code includes an operand in which symbolic reference information [linking information] is inserted in place of an address (Baentsch et al: see [0008], lines 1-15).

Referring to claim 17, Zee/Baentsch discloses the Java execution device of claim 14, wherein the symbolic reference information is inserted during performance of the Ahead-of-Time (AOT) compilation (Baentsch et al: see [0008]-[0009]).

5. Claims 4, 6, 10, 11 and 12 are rejected under 35 U.S.C. 102(e) as being anticipated by US PGPub 2002/0059475 to Baentsch et al (hereafter Baentsch et al) in view of US PGPub 2003/0033344 to Abbott et al (hereafter Abbott).

Referring to claim 4, Baentsch et al discloses a Java class file in a class library which is executed on a Java platform, for use with a Java Virtual Machine (JVM) in which a Java application is executed on a Java platform in a device (see [0005]), comprising:

a constant [constant pools] (see [0008], lines 7-11),

a field [references] (see [0008], lines 6-7), and

a method (see [0008]-[0009]).

While Baentsch et al discloses symbolic linking information (see [0008]-[0009]), Baentsch et al fails to explicitly disclose the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address. Abbott discloses a Java Virtual Machine (see abstract), including the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address (see [0110]).

It would have been obvious to one of ordinary skill in the art at the time of the invention to utilize the step of replacing the address with the linking information as disclosed by Baentsch with reference of Abbott. One would been motivated to do so in order to increase the efficiency of the Java Virtual Machine.

Referring to claim 6, the combination of Baentsch et al and Abbott (hereafter Baentsch/Abbott) the Java class file of claim 4, wherein the symbolic reference information comprises at least one of information on a constant pool symbol (see [0008], lines 7-11), information on a Java Virtual Machine (JVM)-internal symbol and information on a location of a data block.

Referring to claim 10, Baentsch et al disclose a method of improving the performance of a Java platform by precompiling a Java file which is executed on a Java platform in a device, the Java application causing the device to perform a desired function (see [0005]), the method comprising:

converting a Java class file or a Java source file into a machine instruction (see [0007]) including an operand in which symbolic reference information [linking information] is inserted (see [0008]).

While Baentsch et al discloses symbolic linking information (see [0008]-[0009]), Baentsch et al fails to explicitly disclose the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address. Abbott discloses a

Java Virtual Machine (see abstract), including the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address (see [0110]).

It would have been obvious to one of ordinary skill in the art at the time of the invention to utilize the step of replacing the address with the linking information as disclosed by Baentsch with reference of Abbott. One would been motivated to do so in order to increase the efficiency of the Java Virtual Machine.

Referring to claim 11, Baentsch/Abbott discloses the method of claim 10, wherein the Java class file comprises a standard class file [Java standard class files] included in a standard Java class library (see [0007], lines 6-10).

Referring to claim 12, Baentsch et al disclose an execution method of improving the performance of a Java platform in a Java Virtual Machine (JVM) (see abstract) in which a Java application is executed on the Java platform in a device, the Java application causing the device to perform a desired function, the execution method comprising:

determining whether method information of a method to be executed includes an attribute of a code formed of a machine instruction having a operand in which symbolic reference information [linking information] is inserted in place of an address (see [0008]-[0009]); and

if the method information of the method to be executed includes the attribute of the code formed of the machine instruction, linking the symbolic reference information with an address and executing the machine instruction (see [0008]-[0009]).

While Baentsch et al discloses symbolic linking information (see [0008]-[0009]), Baentsch et al fails to explicitly disclose the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address. Abbott discloses a Java Virtual Machine (see abstract), including the further limitation wherein a symbolic reference information indicates a specific class, field or method of an object, and method information of the method comprises an attribute of a code formed of the machine instruction having an operand in which the symbolic reference information [symbolic linking information] is inserted in place of an address (see [0110]).

It would have been obvious to one of ordinary skill in the art at the time of the invention to utilize the step of replacing the address with the linking information as disclosed by Baentsch with reference of Abbott. One would been motivated to do so in order to increase the efficiency of the Java Virtual Machine.

6. Claims 5 and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over US PGPub 2002/0059475 to Baentsch et al in view of US PGPub

2003/0033344 to Abbott et al as applied to claim 4 above, and further in view of US Patent No 6,289,506 to Kwong et al.

Referring to claim 5, Baentsch et al disclose the Java class file, however, Baentsch et al fail to explicitly disclose the further limitation wherein the method information further comprises at least one of exception handling information and information used for garbage collection. Kwong et al disclose a method for optimizing Java performance using precompiled code (see abstract), including the further limitation wherein the method information further comprises at least one of exception handling information [catch and throw exceptions] and information used for garbage collection (see column 5, line 34 – column 5, line 6) in order to increase the efficiency of execution.

It would have been obvious to one of ordinary skill in the art at the time of the invention to utilize exception handling information of Kwong et al with the Java class file of Baentsch et al. One would have been motivated to do so in order to increase the efficiency of execution by quickly handling exceptions.

Referring to claim 13, Baentsch et al disclose an execution method in a JVM, however, Baentsch et al fail to explicitly disclose the further limitation wherein, if the method information of the method to be executed does not include the attribute of the code formed of the machine instruction, the execution method further comprises one of Just-In-Time (JIT) compiling and interpreting the method. Kwong et al disclose a method for optimizing Java performance using precompiled code (see abstract), including the further limitation wherein, if the method information of the method to be

executed does not include the attribute of the code formed of the machine instruction, the execution method further comprises one of Just-In-Time (JIT) compiling and interpreting the method (see column 6, lines 47-58) in order to increase the efficiency of execution.

It would have been obvious to one of ordinary skill in the art at the time of the invention to utilize exception handling information of Kwong et al with the Java class file of Baentsch et al. One would have been motivated to do so in order to increase the efficiency of execution by quickly handling exceptions.

7. Claims 8, 9 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over US Patent No 6,289,506 to Kwong et al as applied to claim 7 above, and further in view of US PGPub 2002/0059475 to Baentsch et al.

Referring to claim 8, Kwong et al disclose pre-compiling a class file, however, Kwong et al fail to explicitly disclose the further limitation wherein step (a) further comprises inserting symbolic reference information. Baentsch et al disclose a java run-time system with modified linking identifiers, including the further limitation of inserting symbolic reference information [references which name target items to which they refer] into an operand of the machine instruction (see [0008], lines 11-15) in order to increase the efficiency of executing the machine code.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use reference linking information disclosed by Baentsch et al as a added feature to the pre-compiling step disclosed by Kwong et al. One would have

been motivated to do so in order to increase the efficiency of executing the machine code.

Referring to claim 9, the combination of Kwong et al and Baentsch et al (hereafter Kwong/Baentsch) discloses the method of claim 8, wherein step (b) further comprises converting the symbolic reference information inserted in the operand of the machine instruction into an address (Baentsch et al: see [0008], lines 11-15).

Referring to claim 16, Kwong et al disclose pre-compiling a class file, however, Kwong et al fail to explicitly disclose the further limitation wherein precompiling the class file comprises inserting symbolic reference information in place of an address into an operand of the machine instruction. Baentsch et al disclose a java run-time system with modified linking identifiers, including the further limitation wherein precompiling the class file comprises inserting symbolic reference information in place of an address into an operand of the machine instruction (see [0008]-[0009]) in order to increase the efficiency of executing the machine code.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use reference linking information disclosed by Baentsch et al as a added feature to the pre-compiling step disclosed by Kwong et al. One would have been motivated to do so in order to increase the efficiency of executing the machine code.

Response to Arguments

8. In regards to applicants arguments concerning the prior art rejection of claim 7, applicant states: There is no teaching or suggestion in Kwong of precompiling a class file included in a standard class library into an extended class library file including a machine instruction as required by claim 7.

The examiner respectfully disagrees. To further clarify, Kwong discloses Java Base Classes 206 and Java Standard Extension Classes 210. The base classes are considered to represent the standard library and the extension classes are considered to represent the extended library. The client system provides the process for compiling the class files (see column 6, lines 29-31).

9. In regards to applicants arguments on pages concerning the prior art rejection of claim 1, applicant states: There is no teaching or suggestion in Zee of an extended class library which includes a class file of a machine code obtained by precompiling a class file included in a standard class library as recited in claim 1.

The examiner respectfully disagrees. Zee discloses downloading a Java class file (see [0060], line 6). The designated class file is considered to represent a file from the standard library. Zee then discloses that "The compile-on-demand server 20 or 22 selects an AOT compiler which can produce a native component for the data processing system ...") This is considered to represent the step of precompiling. Next the native component produced by the compiler is stored in the database. The combination of the file and the native component is considered to represent the extended class library.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kimberly Lovel whose telephone number is (571) 272-2750. The examiner can normally be reached on 8:00 - 4:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Cottingham can be reached on (571) 272-7079. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Kimberly Lovel
Examiner
Art Unit 2167

17 September 2007
kml

JOHN COTTINGHAM
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100